REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE Professional Paper	3. DATES COVERED
4. TITLE AND SUBTITLE CASTLE: The Next Generation of Navy Flight Simulation		5a. CONTRACT NUMBER 5b. GRANT NUMBER 5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Brent York; Thomas Magyar; James Nichols, III		5d. PROJECT NUMBER 5e. TASK NUMBER 5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Air Warfare Center Aircraft Division 22347 Cedar Point Road, Unit #6 Patuxent River, Maryland 20670-1161		8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)		10. SPONSOR/MONITOR'S ACRONYM(S) 11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The Controls Analysis and Simulation Test Loop Environment (CASTLE) aircraft simulation environment has been in development by the United States Navy since 1986, and has become a proven and effective tool used for thousands of hours of piloted simulation each year. CASTLE can be used for such diverse purposes as desktop analysis of flight maneuvers, flight path reconstruction and visualization, pilot-in-the-loop operation in a laboratory environment, and real-time simulation on the desktop. New features are still being added to CASTLE, with improvement to the software being made on a continual basis.

15. SUBJECT TERMS

Controls Analysis and Simulation Test Loop Environment (CASTLE)

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON Brent York / Thomas Magyar / James Nichols
a. REPORT	b. ABSTRACT	c. THIS PAGE		7	19b. TELEPHONE NUMBER (include area code) (301) 757-0853 / 0852 / 0857

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39-18

20010716 016

AIAA-2001-4243

CASTLE: THE NEXT GENERATION OF NAVY FLIGHT SIMULATION

Brent W. York*, Thomas J. Magyar†, James H. Nichols III*

Naval Air Systems Command, Patuxent River, Maryland

Abstract

The CASTLE aircraft simulation environment has been in development by the United States Navy since 1986, and has become a proven and effective tool used for thousands of hours of piloted simulation each year. CASTLE can be used for such diverse purposes as desktop analysis of flight maneuvers, flight path reconstruction and visualization, pilot-in-the-loop operation in a laboratory environment, and real-time simulation on the desktop. New features are still being added to CASTLE, with improvements to the software being made on a continual basis.

Introduction

CASTLE (the Controls Analysis and Simulation Test Loop Environment) is a highly-modular, richly-featured environment for executing aircraft simulations, whether for non-real-time analysis or with a pilot in the loop. The software has been in use and active development by the United States Navy since 1986. Its features and usability have seen continuous and marked improvement during this period. CASTLE is designed for use in stand-alone desktop analysis as well as in real-time piloted simulation. The same airframe model executable is used for both types of simulation, which simplifies simulation development and testing for laboratory and training environments.

Though originally designed primarily for use in a laboratory environment at the Navy's Manned Flight Simulator (MFS) facility[1,2] on Patuxent River Naval Air Station, Maryland, CASTLE has evolved into a suite of tools usable for many applications and on many different computer platforms, from mainframes to laptop PCs. CASTLE is currently being used in high-fidelity aircraft simulation by all branches of the United States Armed Forces, NASA, EADS (Germany), DSTO/AMRL (Australia), and many other government, military, and educational organizations world-wide.

Historically, CASTLE has been utilized primarily in the laboratory at MFS in support of local flight test operations. However, the advent of more powerful personal computers and their ready availability has resulted in a greatly expanded group of CASTLE users, and a revised mission for the software; desktop deployment is now at least as important as laboratory support. In addition, CASTLE is now being more widely used as a part of the architecture for training devices fielded by various branches of the armed forces. Several development paths are being pursued simultaneously for CASTLE, which has experienced rapid change and growth in the past few years, and will continue to do so in the future.

Overview

Five major features characterize the CASTLE environment. CASTLE provides *continuity* of look-and-feel between different aircraft simulations. There is *commonality* between the real-time (pilot-in-the-loop) and analysis modes. The core components of CASTLE represent the *reusability* aspect of the software. Through the use of documented interfaces, the environment affords *extensibility*. Finally, *portability* of CASTLE across computer platforms is provided by careful tailoring of its source code.

CASTLE has been ported to multiple platforms to provide maximum flexibility in the use of existing hardware and to facilitate workgroup interoperability. One set of source code is used to build CASTLE on all supported platforms and operating systems. The same aircraft simulation code can be built, linked with CASTLE, and executed on an SGI™ workstation or on a laptop PC. CASTLE is currently supported on Digital Equipment Corporation AlphaStations running OpenVMS, SGI machines using Irix®, and desktop or laptop PCs utilizing either Linux® or some version of Microsoft® Windows®.

---

* Aerospace Engineer, Senior Member AIAA

† Aerospace Engineer, Member AIAA

in pilot-in-the-loop simulations in laboratory stations at MFS.

The CASTLE source code has changed significantly in the past fifteen years. The original code consisted of both Fortran and Pascal; in 1989, all Pascal code was replaced with Fortran. Much of the code was then converted to C and C++ between 1995 and 1997.

At first, only mainframe computers (specifically, DEC VAX machines) were supported. Support for other platforms was added in the mid-1990s. Originally, the user interface was a simple text-mode menu system. This interface was eventually replaced with one based on the VMS screen management libraries, which provided for better functionality, but was still non-graphical. Finally, in the mid-1990s, a true graphical user interface was implemented.

CASTLE was developed at Manned Flight Simulator, and is used at that facility as frequently as ever. All pilot-in-the-loop simulations at MFS use CASTLE as the airframe simulation environment, and CASTLE is used daily in desktop analysis operations. Increasingly, however, CASTLE is also being used outside MFS, both for analysis and for real-time simulation. Many aircraft simulations, laboratory and desktop, in real-time and analysis mode, U.S. Navy and otherwise, are already being run under CASTLE. Support for desktop simulation has become just as significant as support for real-time simulation in the laboratory in driving CASTLE development. Today, the CASTLE software represents a proven, mature technology that has been used successfully for thousands of piloted simulation sessions, in addition to countless hours of desktop analysis.

Desktop Analysis

The first primary operating mode of CASTLE is desktop analysis. In this mode, the simulation runs as quickly as possible within the limitations of the host hardware (synchronization to wall-clock time is not performed). This type of operation is used for simulation development and verification, for analysis of flight test data, and so on.

CASTLE includes several built-in tools useful in data analysis. Non-intrusive run-time data access is provided through a variable access facility. Variable interrogation and modification can be performed without pausing a running simulation or resorting to the use of a debugger.

The digital data storage and export facility allows any external tool to be used for simulation data analysis. Multiple data sets can be saved at different rates. Data can be output in various formats (ASCII, MATLAB, or user-defined). The number of parameters and the amount of data that can be stored are limited only by memory constraints. In addition, data can be piped directly to MATLAB if installed on the same computer as CASTLE.

An open-source plotting package called XMGR has been modified for use with CASTLE. The GUI is able to spawn XMGR and send data directly to it for inspection of simulation variables. Using the data export functionality, simulation results can instead be plotted using MATLAB if desired.

A linear model extraction (LME) facility is available for the generation of linear state-space models from a nonlinear simulation. The LME utility calculates the A, B, C, and D matrices appearing in the state-space equation

$$\dot{x} = Ax + Bu$$

$$y = Cx + Du$$

The user specifies the states, derivatives, inputs, and outputs; the states and inputs are perturbed and the partials matrices calculated.

Various tools are available for exercising the simulation to extract desired data. A maneuver generation utility is provided, which allows the user to define simple maneuvers such as doublets and sine waves, or to drive the simulation with previously recorded data. A "sweep" facility enables the execution of simple loops over one or more simulation variables, and can be used for such tasks as extracting aerodynamic coefficients or generating trim maps. Macro command files can be used for simple task automation.

Real-Time Simulation

The second major operating mode of CASTLE is real-time simulation. In this mode, the simulation operates in lock step to wall-clock time. Synchronization information can be obtained from an external source (such as a sync pulse from a visual image generator), or can be provided by the host computer itself. This mode is used for pilot-in-the-loop simulation.

Real-time simulation in a laboratory or trainer environment is typically performed using a mock-up of an actual aircraft cockpit as the pilot input device. CASTLE also supports real-time simulation on the desktop through the use of CasView, with simple controls (usually joystick, throttle, and rudder pedals) taking the place of the more traditional cockpit setup.

Figure 2 illustrates the process intercommunication that takes place when running CASTLE with a pilot in the loop. Unshaded boxes represent CASTLE components, while shaded boxes stand for processes or hardware external to CASTLE. All components communicate either through DTM (TCP/IP) or through shared memory.

The shared memory support provided by CASTLE is encapsulated in a library called VIO (Variable Input/Output). Support for various methods of accessing shared memory is provided in VIO, and the library is extensible to allow the addition of new shared memory types in the future. For single-machine arrangements, shared memory resides in local RAM. For simulations using multiple computers, shared memory represents some type of memory accessible by all machines, such as dual-port memory or ScramNet.

Shared memory contents are accessible to outside applications for further processing, which provides the opportunity to interface user-provided external software with the simulation. Avionics, network interfaces, and other such processes are often connected to the aircraft simulation in this manner.

Simulation execution in real-time can optionally be controlled by an instructor/operator station, cockpit control panel, or other user-supplied control program, instead of relying on the CASTLE GUI. The communications protocol between the GUI and the airframe model is documented, to allow designers of external processes to incorporate CASTLE airframe model commands into their own software.

CasView

The desktop flight path visualization tool included with CASTLE is called CasView. This software provides an out-the-window display, aircraft instruments, and a pilot inputs monitor. CasView can be used to display flight test data, to monitor real-time operations, and to provide an inexpensive desktop real-time flight environment. Because of the machine-specific nature of three-dimensional graphics optimization, CasView is currently only supported on the Microsoft Windows family of operating systems. Figure 3 displays sample screenshots from the CasView utility.

CasView provides the option of using simple game-type controllers for pilot inputs. The signals from these controllers are passed to CASTLE in the same manner as signals from a full-fidelity cockpit. This option permits the desktop user to fly the simulation as though it were running in the laboratory or trainer environment. This arrangement is particularly useful for testing simulation modifications, or attaining flight conditions not easily reproducible using simple pilot input models. Although desktop real-time operation is extremely useful for debugging and for certain data analysis operations, it should always be remembered that the flying qualities of the simulation will be adversely affected by using simple game-type controllers, instead of a more accurate model of the controls as is often provided in a higher-fidelity laboratory or trainer environment.

Extensive support for aural cueing is provided. The user can import any Windows WAV file into CasView, to be played whenever specified conditions are met. Such conditions are set based on data from the aircraft simulation, and can be used for normal aircraft aural

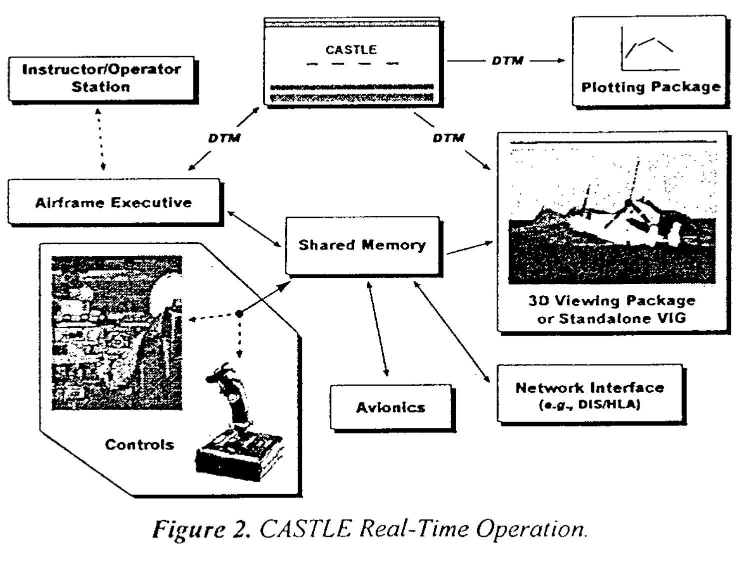

Figure 2. CASTLE Real-Time Operation.

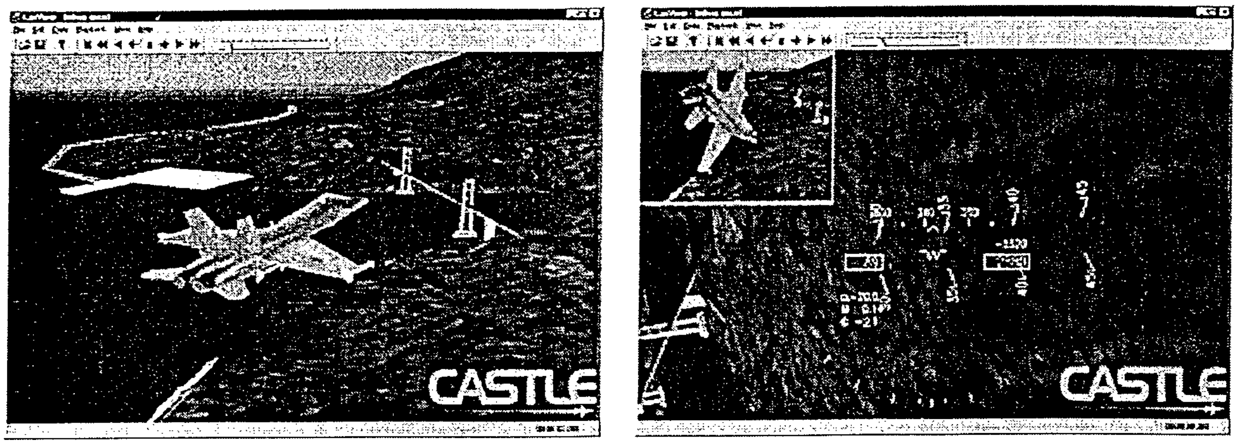

Figure 3. *CasView screenshots.*

cueing (such as an altitude voice warning or a stall tone) or environmental sound effects (such as engine noise or canopy rumble).

CasView includes a documented interface for arbitrary user-defined plug-ins. These plug-ins are Windows DLLs that can be created using any appropriate Windows programming environment (such as Microsoft Visual C++). Four types of plug-ins can be created for and used in CasView: head-up displays, deformable ownships, instrument panels, and arbitrary objects. A head-up display (HUD) plug-in represents the display elements projected in front of the pilot. A deformable ownship plug-in controls changes to the ownship (the three-dimensional model representing the pilot's aircraft) in response to data; for example, it can move the model's control surfaces or spin its rotors. Instrument panel plug-ins represent lights, gauges, and other displays within the cockpit. Arbitrary object models represent any three-dimensional entities the user wishes to insert into the out-the-window scene; examples of such objects include velocity vectors, projected flight path markers, aircraft carriers, or even other aircraft. Only one unique HUD plug-in and one deformable ownship plug-in can be loaded into CasView at a time, but any number of instrument panel and arbitrary object plug-ins may be loaded and active at any given time.

CasView is distributed with a simple checkerboard terrain model as well as a basic land and sea terrain database. If desired, users can import their own terrain models. CasView currently supports the use of any OpenFlight-format terrain database for the out-the-window display.

A CasView run can be stored as a movie for later viewing without requiring the use of the CasView utility itself for playback. These movies are output as Windows AVI files, which can be played by various readily available utilities, or inserted into PowerPoint presentations.

Because CasView communicates with CASTLE through shared memory, it is a simple matter to extend the desktop real-time environment to add support for external processes such as avionics or auxiliary displays. For example, a software model of a proposed avionics unit could be configured to read and write data from the same shared memory area as CASTLE and CasView, enabling it to sense the aircraft state and to display any desired output through CasView. In fact, this shared memory interface has been used in the past at MFS to connect a CASTLE/CasView desktop real-time session directly to an avionics unit for hardware-in-the-loop testing.

Future Work

While great strides have been made in improving CASTLE, much work still must be done for CASTLE to remain viable going into the twenty-first century. Many new features and improvements are currently under development.

CasView Improvements

Several additional features are being investigated for the CasView utility. New methods for data visualization within CasView, such as strip charts and data spreadsheet views, are being developed. Alternative visualization schemes, using multiple monitors or head-mounted displays, are being investigated for use in both laboratory environments and desktop visualization.

Improvements in the handling of custom terrain are being made. The size of a terrain database is currently

limited by available mainboard and video card memory; the ability to page in pieces of large databases as required is being developed. The feasibility of generating a simple terrain database automatically from supplied real-world elevation data is also being investigated.

The ability to reconstruct missing data channels is desired. Data sets originating from flight test often omit some basic channels, but include all the information necessary to deduce the missing data. For example, a data set might contain the aircraft body axis velocities, the Euler angles, and the angles of attack and sideslip, but no position or inertial velocity data. CasView should be able to reconstruct these channels (and apply kinematic consistency checks) without user intervention. Eventually, it is desired to support the direct read of flight recorder data in the native file format, for a quick-turnaround post-flight visualization of aircraft behavior.

Redesigned GUI and Toolbox

A modern cross-platform user-interface for both CASTLE proper and its associated tools is under development. The currently-used X-Windows/Motif-style GUI is not particularly intuitive for users of the Microsoft Windows family of operating systems. Because Windows is becoming the operating system of choice for the majority of CASTLE desktop users, it is important for user acceptance to provide an interface that is as comfortable and familiar as possible. In addition, X server software (such as Hummingbird™'s Exceed™) must be utilized in order to run the GUI software on a Windows-based system. The X server software adds a unnecessary level of complexity to the GUI process which has been shown to introduce bugs and reduce the perceived reliability of CASTLE.

The new GUI will be Windows-native, with ports to other operating systems to be performed at a later date. It will employ the standard layouts and conventions used in the current generation of Windows software, while making some improvements upon the existing GUI in terms of reliability and user-friendliness. The use of X server software will be obviated by this development, further increasing the reliability of the system.

The simulation development tools included with CASTLE are also being redesigned, with user-friendliness and robustness foremost among the software requirements. These tools were originally developed for VMS or SGI Irix, both command-line-oriented operating systems. They will require some modification to achieve a form with which users of graphically-oriented operating systems like Microsoft Windows will be comfortable.

MATLAB® and Simulink® Interface

An interface between CASTLE and MATLAB and Simulink (products of The MathWorks, Inc.) has been developed[4] and is being expanded. The MATLAB interface lets the engineer control CASTLE directly from the MATLAB environment, which can improve and expedite processes such as flight test data analysis. If engineers prefer to do the majority of their work in MATLAB, they can remain within that environment and still gain the benefits of using an existing high-fidelity aircraft simulation hosted under CASTLE. The Simulink interface will allow rapid development of control systems or other aircraft system models while providing the engineer with the ability to use a high-fidelity simulation of the remainder of the aircraft in the Simulink loop.

CASTLE.BASIC

One of the greatest needs for CASTLE is a modern scripting language. CASTLE currently supports a rudimentary command file structure, which allows the user to list in an ASCII text file commands representing selections on the CASTLE menus and dialogs. However, this format does not include any provision for process control flow (looping and branching), for variable creation, and so on. A scripting language called CASTLE.BASIC is being developed which will address these shortcomings. The aim of this effort is to provide the user with a powerful tool to facilitate the automation of repetitive tasks, such as running a suite of database validation tests. The current system is certainly usable (and is frequently used) for such an operation, but CASTLE.BASIC will make the associated script files easier to create and to maintain.

Trainer Support

It is a current design goal of CASTLE to provide a high level of usability to the trainer community. Standardization of trainer architectures is a desirable goal, and commonality of trainer software architecture with that of the laboratory used in development of aircraft models is particularly important. CASTLE is already in use in the development and validation of aerodynamic databases and flying qualities models for many U.S. Navy aircraft; distributing validated simulations to trainers could be made simpler by incorporating CASTLE into the trainer architecture.

To that end, "drop-in" usability for trainer development is being improved through the addition of trainer-

specific support modules. Standardized automated fidelity testing methodologies are being investigated and incorporated. Various modifications are being made to the core CASTLE code to address more fully the unique requirements of the trainer community.

Networked Desktop Simulation

Networked simulation on desktop and laptop computers is being investigated, for the purpose of supporting large multiplayer simulations for training purposes. CasView was originally developed for flight path visualization, but can also be used as an inexpensive out-the-window display for real-time networked simulation. Modifications are being made to CASTLE and to CasView to allow multiple users on separate computers to fly interactively in the same virtual world. Because complex aircraft models can easily be run in real time on modern laptop computers, the applications for this technology in a training environment are limitless.

In addition, an effort is underway to make the airframe model framework "HLA-aware." The intent is to simplify the insertion of CASTLE objects into existing networked warfare scenarios. These objects could be human-controlled, or they could be automated entities.

Other next-generation simulation tools and features are also in development. In summary, the CASTLE environment is being continually improved and updated in order that it may continue to serve the needs of the aircraft simulation community well into the future.

Conclusion

CASTLE is a flexible simulation environment that operates on multiple platforms and operating systems, and is designed for the desktop as well as the laboratory or trainer. It is currently being employed on a daily basis by a wide range of simulation users in a variety of situations. CASTLE incorporates fifteen years of user-requested capabilities, and encompasses a broad range of sophisticated development, analysis, and validation tools.

---

[1] Nichols, James, Thomas J. Magyar, and Eric C. Schug, "The Platform-Independent Simulation Environment at Manned Flight Simulator," AIAA-98-4179, 1998.

[2] Nichols, J.H., "The Generic Simulation Executive at Manned Flight Simulator," AIAA-94-3429, 1994.

[3] McFarland, R., "A Standard Kinematic Model for Flight Simulation at NASA-Ames," NASA CR-2497, January 1975.

[4] Magyar, Thomas J., and Anthony B. Page, "Integration of the CASTLE Simulation Executive with Simulink," AIAA-2001-4121, 2001.